Accurate travel time prediction with state-space neural networks under missing data

Article *in* Transportation Research Part C Emerging Technologies · October 2005

DOI: 10.1016/j.trc.2005.03.001

3 authors:

J.W.C. Van Lint
Delft University of Technology
164 PUBLICATIONS **2,103** CITATIONS
SEE PROFILE

Serge Hoogendoorn
Delft University of Technology
471 PUBLICATIONS **6,364** CITATIONS
SEE PROFILE

Henk van Zuylen
Delft University of Technology
181 PUBLICATIONS **2,415** CITATIONS
SEE PROFILE

Available online at www.sciencedirect.com
SCIENCE DIRECT®
Transportation Research Part C 13 (2005) 347–369
TRANSPORTATION RESEARCH PART C
www.elsevier.com/locate/trc

Accurate freeway travel time prediction with state-space neural networks under missing data

J.W.C. van Lint *, S.P. Hoogendoorn, H.J. van Zuylen

Delft University of Technology, Faculty of Civil Engineering and Geosciences, Department of Planning and Transportation, P.O. Box 5048, 2600 GA Delft, The Netherlands

Received 12 June 2003; accepted 16 March 2005

Abstract

Accuracy and robustness with respect to missing or corrupt input data are two key characteristics for any travel time prediction model that is to be applied in a real-time environment (e.g. for display on variable message signs on freeways). This article proposes a freeway travel time prediction framework that exhibits both qualities. The framework exploits a recurrent neural network topology, the so-called state-space neural network (SSNN), with preprocessing strategies based on imputation. Although the SSNN model is a neural network, its design (in terms of input- and model selection) is not "black box" nor location-specific. Instead, it is based on the lay-out of the freeway stretch of interest. In this sense, the SSNN model combines the generality of neural network approaches, with traffic related ("white-box") design. Robustness to missing data is tackled by means of simple imputation (data replacement) schemes, such as exponential forecasts and spatial interpolation. Although there are clear theoretical shortcomings to "simple" imputation schemes to remedy input failure, our results indicate that their use is justified in this particular application. The SSNN model appears to be robust to the "damage" done by these imputation schemes. This is true for both incidental (random) and structural input failure. We demonstrate that the SSNN travel time prediction framework yields good accurate and robust travel time predictions on both synthetic and real data.

* Corresponding author.
E-mail address: h.vanlint@citg.tudelft.nl (J.W.C. van Lint).

0968-090X/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.trc.2005.03.001

Keywords: Advanced traffic information systems; Recurrent neural networks; State-space neural networks; Robustness; Missing data; Simple imputation; Non-linear state-space models; Traffic prediction; Travel time prediction; Neural network design

1. Introduction

There is an increasing need for Advanced traffic information systems (ATIS's) that can provide road-users and traffic managers with accurate and reliable real-time traffic information (Abdel et al., 1997). The potentially beneficial effects of ATIS's (in terms of individual and collective cost- or time savings) have been studied extensively in the past decade (e.g. Khattak et al., 1995; Arnott et al., 1991; Mahmassani and Liu, 1999). Although these studies clearly show the heterogeneity of possible responses to ATIS among different groups of drivers (e.g. commuters, non-commuters) under different (traffic) circumstances, they generally do emphasize two things. The first is that for traffic information to have beneficial effects, it should be based on *predictions* rather than on current or past traffic conditions (Chen et al., 1999). Secondly, the *reliability* of traffic information greatly influences driver response (Mahmassani and Liu, 1999; Van Berkum and Van der Mede, 1993). In a real-time setting, the reliability of traffic prediction models is among other things a function of the sensitivity of the models used to unreliable and/or incomplete input data.

In this article we present a neural network based travel time prediction framework that is both accurate and capable of dealing with (i.e. robust to) missing or corrupted input data. The model is aimed at short-term travel time prediction on freeways, typically for generating messages (expected travel times) on variable message signs (VMS's) on strategic locations (e.g. bifurcations) on a freeway network, or for real-time en-route travel information services such as RDS-TMC.[1] The framework consists of a recurrent neural network travel time prediction model, in conjunction with a preprocessing layer which utilizes simple imputation schemes to remedy input failure.

The article is outlined as follows. In the next section we will give an overview of the state-of-the-art in freeway travel time prediction. Thereafter, we present a novel recurrent neural network topology specifically designed for freeway travel time prediction: the so-called state-space neural network (SSNN). Subsequently, we will address the robustness of the SSNN model under missing or faulty input data and show results of the SSNN travel time prediction model on both synthetic and real data.

2. The freeway travel time prediction problem: state-of-the-art

Travel times on freeways are (certainly in congested conditions) the result of complex traffic processes, which are governed by stochastic and non-linear interactions between individual drivers. Evidently, *predicting* travel times for vehicles departing on a particular freeway route is as complex as predicting the underlying traffic processes vehicles will encounter, which—based on

[1] Radio data signal—traffic message channel.

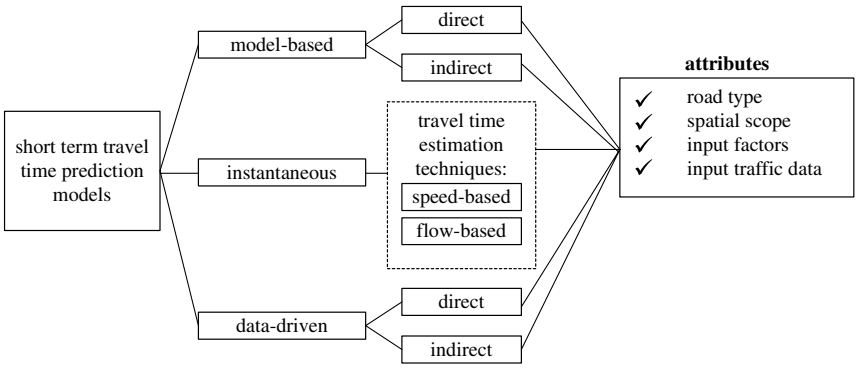

Fig. 1. Taxonomy for short-term freeway travel time prediction.

five decades of traffic flow theory (e.g. Hoogendoorn and Bovy, 2001)—is generally accepted as a challenging problem. As outlined in Fig. 1 we identify three basic strands of approaches to the short-term freeway travel time prediction problem. These are direct and indirect[2] model-based approaches, instantaneous approaches and (direct or indirect) data-driven approaches.

The first class of model based travel time predictors solve the travel time prediction problem by predicting traffic conditions for a sufficient number of time periods ahead[3] and then subsequently deduce mean travel times from these predicted traffic conditions. Examples include DynaMIT (Ben-Akiva et al., 2002) and DynaSMART (Hu, 2001), METANET (e.g. Van Grol et al., 1997; Smulders et al., 1999). The second strand of travel time prediction approaches involve so called *online* or *instantaneous* travel time *predictors*.*Travel time estimation* pertains to deducing travel times from "known" (i.e. historic) traffic conditions such as speeds or flows (see e.g. Van Lint and Van der Zijpp, 2003). Inherently, travel time estimators can only provide estimates of past travel times. If it is assumed, however, that current traffic conditions remain stationary for an indefinite time period, then travel time estimates based on these stationary conditions can serve as travel time predictions. These models are referred to as instantaneous travel time predictors, see e.g. Zhang and Rice (2003) and Van Toorenburg (1998)—in Dutch. Finally, a wide range of data-driven models have been applied to predict both traffic flows and speeds as well as travel times on road networks. These approaches include ARIMA(X) models (Williams, 2001), non-linear time series modeling (Ishak and Al-Deek, 2002; D'Angelo et al., 1999) linear (Rice and Van Zwet, 2001; Zhang and Rice, 2003; Sun et al., 2003), and support vector regression models (Chun-Hsin et al., 2003); feed-forward (Innamaa, 2001; Rilett and Park, 2001; Park and Rilett, 1999; Cheu, 1998; Huisken and Van Berkum, 2003) and recurrent (Abdulhai et al., 1999; Dia, 2001; Ishak et al., 2003; Van Lint et al., 2002) neural networks and various hybrid approaches (Chen et al., 2001; Park and Rilett, 1998; Ishak and Alecsandru, 2003), which for example use neuro-fuzzy

[2] *Indirect* travel time prediction pertains to predicting other quantities (e.g. flows or speeds) which are subsequently used to derive predicted travel times. There is a chicken-and-egg complexity here, since the minimally required time window for predicting speeds or flows in fact equals the travel time we wanted to predict in the first place. Practically, one would predict speed (or flow) for each location (or section) as much into the future (rolling horizon) as required for the longest (probable) occurring travel time.

[3] The exact number of time periods off course equals the travel time which one wanted to predict in the first place.

approaches, or combinations of different ANN topologies. All these data-driven methods have in common that they correlate mean (observed) travel times or traffic conditions to current and past traffic data, without explicitly addressing the (physical) traffic processes that determine these travel times as model based approaches do. The clear advantages of data-driven approaches in general are that they do not require extensive expertise on traffic flow modeling, many ready-to-use software packages are available for model design and calibration, they are fast and easy to implement, and specifically neural network approaches have proven accurate and reliable traffic predictors (Dougherty, 1995). There is, however, one major drawback from which all data-driven approaches, including sophisticated neural networks, suffer. They are all *location-specific* solutions, requiring significant efforts in input- and model selection for each specific application, via for instance correlation analysis, or genetic algorithms or trial-and-error procedures. Results obtained for one location are (typically) not transferable to the next, due to location-specific circumstances (geometry, traffic control, etc.). Our approach aims at combining the advantages of model-based approaches with those of data-driven and specifically neural network approaches, that is straightforward and traffic related design, combined with ease-of-implementation and accuracy.

3. The state-space neural network (SSNN) for freeway travel time prediction

3.1. Derivation

Since traffic is characterized by complex patterns over both space and time, we seek a neural network topology capable of simulating the evolution of spatiotemporal input–output mappings. Since we also require that the topology of this neural network should be derived from traffic-related considerations rather than from black box approaches such as genetic algorithms or correlation analysis we base our efforts on a so-called discrete state space model (DSSM). A DSSM contains a dynamic equation governing the state dynamics and an output equation that maps the current states to the model output:

$$\mathbf{x}(t) = F(\mathbf{x}(t-1), \mathbf{u}(t), \mathbf{V}) \text{ [state equation]} \tag{1a}$$

$$y(t) = G(\mathbf{x}(t), \mathbf{w}) \text{ [output equation]} \tag{1b}$$

with $\mathbf{x}(t) = \{\ldots, x_m(t), \ldots\}$, $m \in \{1, \ldots, M\}$, where M depicts the total number of adjacent sections comprising route r. As is shown in (1a) and (1b), the state $\mathbf{x}(t)$ of the sections on time instant t, is uniquely defined by the previous state-vector $\mathbf{x}(t-1)$ and the section specific input vectors $\mathbf{u}(t) = \{\ldots, \mathbf{u}_m(t), \ldots\}$ of the time period $[t-1, t]$, which contain speeds and flows from measurement locations on the sections, and in- and outflows from on- and off-ramps, connecting to the sections.[4] Note that the state $x_m(t)$ of a single section depends not only on its own previous state

[4] Note that the state dynamics are sometimes defined as $\mathbf{x}(t) = F(\mathbf{x}(t-1), \mathbf{u}(t-1), \mathbf{V})$, in which case $\mathbf{u}(t-1)$ depicts the input obtained in time period $[t-1, t]$.

but also on the previous state of other sections along the route of interest. This reflects the effect of state information flowing in both up- and downstream directions, dependent on which traffic conditions prevail (free-flowing or congested). The output function, in this case a scalar function $y(t)$, calculates the mean travel time for a vehicle starting the route at time instant t, and takes a vector $\mathbf{x}(t)$ of all section states at time instant t as inputs. Finally, $\mathbf{V}$ and $\mathbf{w}$ denote parameter vectors associated with the dynamic Eq. (1a) and output Eq. (1b) respectively, which both can be adjusted during calibration.

Given an appropriate choice of functions F and G (see below) this DSSM is in fact a One Layer First Order Context (FOC) Memory in the taxonomy of Kremer (2001)—similar to the Elman RNN proposed in Elman (1990). Also Dorffner (1996) showed that an Elman network in fact is a specific realization of a general DSSM. We will refer to it from here on as the state-space neural network (SSNN), see Fig. 2. The signals produced by the hidden layer will be referred to here after as the *internal states* of the SSNN. The context layer effectively provides a short term memory for these internal states. Mathematically, the SSNN works as follows. A neuron j in both hidden and output-layer calculates its output z_j as a weighted sum of its inputs u_{ji} and a bias w_{j0}

$$z_j = w_{j0} + \sum u_{ji} w_{ji} \tag{2}$$

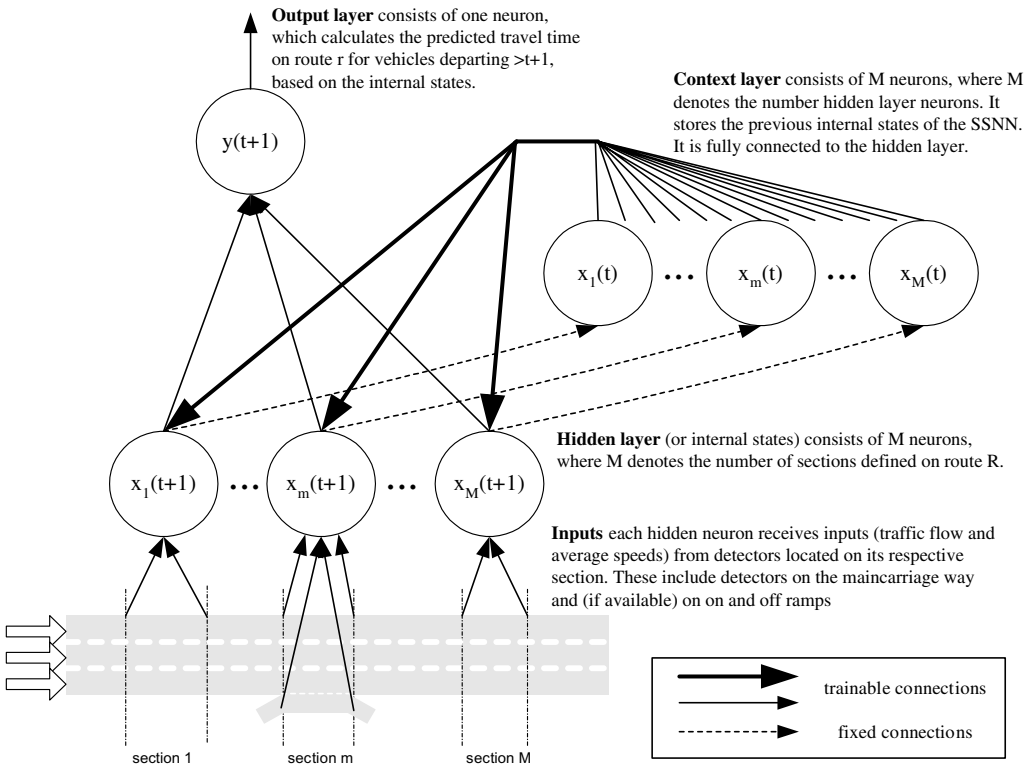

Fig. 2. State-space neural network (SSNN) topology for short-term freeway travel time prediction.

Subsequently, the output is transformed by the well-known sigmoid transfer function

$$\phi(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

which can be written in matrix form $\Phi(\mathbf{z}) = (\phi(z_1), \phi(z_2), \ldots, \phi(z_M))^T$. Using matrix notation the SSNN can now be written in the same state space form as the generic DSSM model (Eqs. (1a) and (1b)) and reads

$$\mathbf{x}(t) = \Phi(\upsilon_0 + \upsilon\mathbf{u}(t) + v\mathbf{x}(t-1)) \tag{4}$$

$$y(t) = \phi(\omega_0 + \omega\mathbf{x}(t)) \tag{5}$$

where υ_0, υ, and v denote the bias and weight vectors associated with the hidden layer (comprising $\mathbf{V}$), $\mathbf{u}(t)$ denotes a concatenated vector of all section specific input vectors, and ω_0 and ω the bias and weight vector (comprising $\mathbf{w}$) of output layer. The choice of a non-linear (logistic) output function is arbitrary, a linear function would also suffice. Summing up all inputs (speeds, flows) and internal states implicates that we cannot assign a direct physical quantity to the hidden neuron outputs (the internal states). At best we could interpret each hidden neuron's output $x_m(t)$ as a indicator for the amount of delay (travel time) on section m at time $(>)t$.

One of the central issues in applications of neural nets to for example traffic prediction is in choosing the appropriate structure in terms of the number of hidden neurons and the degree of connectivity between layers. Bishop (1995) devotes an entire chapter (9) to the issue of model complexity (number of parameters and hidden units) in the light of generalization (how well does the model perform on unseen data). Too few (parameters) lead to biased and inadequate models, too many to overly complex models that generalize poorly. There exist universal theorems that prove a two or three-layered neural net can approximate any non-linear mapping, given sufficient hidden neurons, but these have very little practical value for model designers (Hecht-Nielsen, 1990; Bishop, 1995). Most rules of thumb do not provide much structural value in choosing the number of hidden neurons either. As Warren Sarle (Sarle, 2004) convincingly argues: "These rules of thumb are non-sense because they either ignore the number of training cases, the amount of noise in the targets, and the complexity of the function. Even if you restrict consideration to minimizing training error on data with lots of training cases and no noise, it is easy to construct counterexamples that disprove these rules of thumb". In our case we start with an educated and application-specific "guess", that is, as much hidden neurons as we have defined sections on our freeway. More over, we specifically choose to connect neurons with section specific inputs only, in analogy with traffic flow models. Hidden and context layer, however, are fully connected, so that each hidden neuron receives all previous hidden neurons activities. This is based on the notion that each internal state may be influenced by what happened previously elsewhere on the route. Although these choices may be considered arbitrary in a neural network context, training this specific topology with the Bayesian regularized Levenbergh–Marquardt algorithm consistently leads to an accurate model (on test data) and a parameter setting in which no more than 25% of the parameters are dubbed effective, which indicates that the model is accurate and valid and has more than enough hidden neurons/parameters to solve the problem. One of the mathematically most sound methods of neural network complexity control is Bayesian regularization (BR), based on the seminal work of MacKay (1995), which is the method we use and recommend.

3.2. Training

In general terms, the SSNN can be trained in a supervised manner to approximate any mapping $y(t) = g(w, \mathbf{u}(t-1))$, if sufficient and representative data pairs $\{\mathbf{u}(t-1), o(t)\}$ are available, where $o(t)$ denotes the output of the real process (expected travel time for a vehicle starting at t) given input pattern $\mathbf{u}(t-1)$. In this context $\mathbf{w}$ denotes a vector containing all adjustable parameters (weights and biases) in the model. The task of learning then is to find a set of adjustable parameters $\mathbf{w}$, which gives a mapping that fits the training-set well, and is capable of generalizing well to "new" data (MacKay, 1995). This can be achieved by minimizing a cost function $F(\mathbf{w})$ with a regularization term:

$$F(\mathbf{w}) = \beta E_D + \alpha E_W = \beta \sum_T \frac{1}{2}(y(t) - o(t))^2 + \alpha \sum_W \frac{1}{2} w_i^2 \tag{6}$$

The parameters β and α regulate to which extend the output error (the first term in Eq. (6)) and the size of the weights (the second term) contribute to the performance function. Regularization takes into account that increasing model complexity (larger or more weights) may lead to better performance on the training data, but also to poorer generalization with respect to unseen data. Details on the algorithms used (Levenberg–Marquardt and Bayesian Regularization LM–BR) can be found in Hagan and Menhaj (1994) and Foresee and Hagan (1997) respectively.

Note that although the SSNN is a recurrent neural structure, it can—like the Elman structure—be trained in a *feed-forward* manner by truncating the temporal dynamics at time $t-1$. By doing so the SSNN model can essentially be unfolded as a feed-forward structure and a standard technique can be used for training (in our case LM–BR). For example, Dorffner (1996) and Elman (1990) show this is an appropriate approach, which has also been widely applied in commercial neural network software (e.g. Demuth and Beale, 1998). Although both gradient of the performance function and Jacobian of the output errors are now approximations and there may be limitations in memory depth in case the process is strongly non-Markovian (Hochreiter and Schmidhuber, 1997), the algorithm does yield stable results, certainly throughout our experiments. In our view, the requirements are proper weight initialization (e.g. Nguyen and Widrow, 1990), and adequate complexity control (Bayesian regularization—MacKay, 1995).

4. Behavior of the SSNN under missing data

The input data in a real-time situation, collected by a real-time traffic monitoring system, will often consist of corrupted or missing values (e.g. on average 15% of the inductive loops of the Dutch freeway monitoring system (MONICA) may be out of operation or producing unreliable measurements[5]). Before the SSNN model can be applied in a real-time environment, we need to study its behavior under missing or unreliable input data. To this end we propose a travel time prediction framework (Fig. 3) which consists of two key components: a preprocessing layer, which processes and potentially cleans the data it receives from a real-time traffic detection system, and

[5] Statistic from a week of 1 min aggregate measurements of inductive loop detectors on the A13 highway between Den Haag and Rotterdam, january 2002; 9746 of 65,536 measurements were classified unreliable (missing or faulty) = 14.9%.

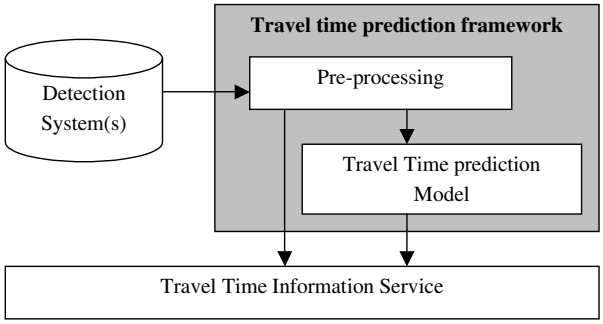

Fig. 3. Robust freeway travel time prediction framework.

the actual travel time prediction model (the SSNN model presented above), which is fed by the preprocessing layer and predicts travel time on the route of interest.

The aim of the strategies discussed in this article is to provide for a "graceful degradation" of performance in case of increasing amounts of missing or corrupted data. In principle, the data cleaning/preprocessing layer performs the following tasks:

1. *Data checking*: before possible problems (e.g. missing data) can be adequately tackled, they need to be detected first.
2. *Data completion*: filling the possible gaps in the data with reasonable replacements.
3. *Data correction*: recheck the now complete data set for validity and consistency and replace/adjust data if required.

In this article we deal with the last two points, that is, and assume it is known that a certain datum is either missing or unreliable/inconsistent. In case of the MONICA detection system, which we will describe later, this is indeed the case.

4.1. Classification and representation of input failure

In this article we use the following definition for input failure:

Definition 1. Input failure is the occurrence of unreliable or missing data in the input vector. This happens when a measurement device produces data that is (either by the modeler or the device itself) dubbed unreliable, or when it produces no data at all.

In the ensuing we interchangeably use the terms input failure and detection (or detector) failure, both adhering to the definition above. For experimental purposes we propose a classification of input failure as presented in Fig. 4. Note that in practice all three types may occur simultaneously. The first type of detection failure (Fig. 4(a)—incidental (random) failures) occurs due to, for example, temporary power or communication failures in the freeway monitoring system. The second type (Fig. 4(b)—structural failures) occurs mainly due to physical damage or maintenance backlogs to the inductive loops or roadside equipment. Although the distinction presented here may not be as crisp in practice, the proposed distinction expresses two extreme configurations of input failure that can be expected in practice, and is hence useful in the investigation of the

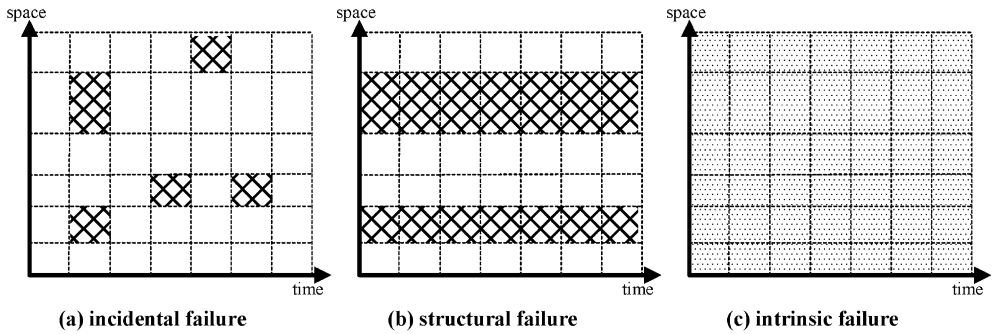

Fig. 4. Classification of possible input failure (i.e. missing or unreliable data from traffic detectors). In practice a mixture of all types of failure will occur.

robustness of travel time prediction models to input failure. A third type of failure (Fig. 4(c)—intrinsic failure), measurement noise and bias, is inherent to detection devices and averaging measurements over time in general. An example of the latter type of input failure, which is discussed in Van Lint and Van der Zijpp (2003) is the fact that in MONICA (an inductive loop based traffic data collection system) the arithmetic time mean speed per measurement period is calculated, yielding a biased estimate of the space mean speed (the mean speed on a particular section), which is in fact the quantity of interest. Other known sources of intrinsic data corruption are miscounts, double counts or false counts of vehicles, device calibration errors, round off errors, etc.

4.2. Strategies for dealing with missing data

We will test the data cleaning strategies as shown in Table 1. Strategies S0 depict null replacement strategies in which we train the SSNN model with increasing amounts of corrupted data. The other strategies (S1–S3) use a SSNN model trained with 100% clean data and simple imputation strategies to clean the input data on beforehand. In the next two subsections these methods are briefly explained.

4.2.1. Null replacement: training the SSNN with missing data

The first strategy (S0 in Table 1) is to leave missing values as is. Since the SSNN has a (a priori) fixed topology, it can only accept input in this (a priori) defined format. This implies that missing data values should be replaced by some default *null* value, which allows the SSNN to detect it as such and distinguish it from valid measurements, while still produce reasonable output, such that "graceful degradation" occurs for increasing amounts of missing data.

Table 1
Strategies for dealing with missing data for the SSNN travel time prediction framework

Strategy	Corruption train data (%)	Corruption test data (%)	Imputation scheme (%)
S0	0–40	0–40	Null
S1	0	0–40	Interp
S2	0	0–40	MA
S3	0	0–40	MA/Interp.

The choice here is made to replace all missing values with 0.5. To illustrate this particular *null* value does lead to a "graceful degradation" of performance, recall the sigmoid transfer function of the previous section, which every hidden neuron in the SSNN uses to transform its input vector $\mathbf{z} = [z_1, \ldots, z_N]$, $z_j \in \langle 0.1, 0.9 \rangle$, $\forall j$ to the scalar value $h \in \langle 0, 1 \rangle$

$$h = \frac{1}{1 + \exp\left(w_0 + \sum_{i=1}^{N} w_i z_i\right)}$$

Fig. 5 shows the neuron output h as a function of the number of missing values in the input vector (which we arbitrarily set to length 9) for four different *null* values, -100, -1, -10^{-6}, and 0.5. Each of the four graphs shows 10 different neuron responses for increasing numbers of *null* values, based on 10 different weight settings, chosen randomly from a zero mean normal distribution with unity variance. Clearly, Fig. 5 (top-left) shows that a large *null* value (e.g. -100 or $+100$) leads to oscillatory behavior even at small percentages of missing data. A small but significant (of the same order as $\mathbf{z}$) negative value of -1 (Fig. 5 (top-right)) does not lead to oscillation, but certainly to an unstable response, since it effectively steers the sigmoid function in the

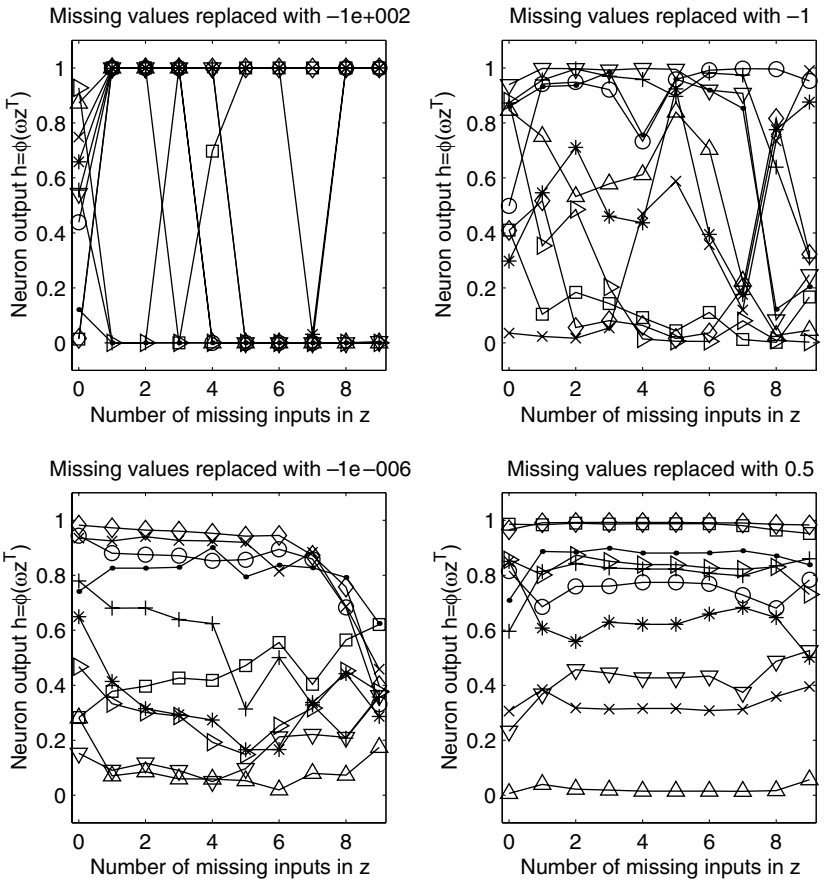

Fig. 5. Effect of setting missing data values to different *null* values on a single neuron with a logistic transfer function. Each graph shows 10 responses representing 10 random weight settings.

wrong direction (either positive or negative depending on the weight setting). A small value (around zero) does improve stability, but still leads to an unwanted neuron response. In case of the SSNN particularly, near zero values represent either very low (or even slightly negative) speeds or very low or zero flows. Since both flow and speed from one detector are replaced with *null* in case the detector fails, the resulting *null* value represents a very unusual and extreme input-signal.[6] Choosing a *null* value in the neurons input domain (0.5, which is the mean of the minimum and maximum input values) as in Fig. 5 (bottom-right) yields by far the most stable result. The value 0.5 represents a "mean" input-signal (moderate flows, moderate speeds), that allows for a graceful deterioration of neuron output. In the extreme case when all input-signals are missing, the neurons' response will equal the mean response it has learned from the training data.

We will train the SSNN with data sets corrupted with a "mix" of both incidental and structural failure. First, five training data sets are generated with an increasing percentage of incidental (random) failure (0%, 10%, 20%, 30% and 40% respectively). Next, in each of the four corrupted training data sets each detector is "shut down" for longer periods of time. Arbitrarily, we choose structural failures of 30 consecutive periods (30 min). At any time instant no more than two detectors are structurally failing. Effectively, this causes an increase of the total percentage of input failure in each training data set of 1–2%. The hypothesis is that the combined effect of incidental (random) and structural failure should enable the SSNN model to develop a parameter setting that implicitly incorporates both types of detector failure.

4.2.2. Simple imputation: interpolation and exponentially moving average

In practice the most commonly used approach to remedy input failure is "simple" imputation. Simple here depicts substituting missing values with sensible ad-hoc replacements, such as regression forecasts or sample mean. Schafer (1997) shows that simple imputation schemes tend to change the covariance structure of the input-data and may induce bias. Therefore, Schafer proposes EM and Markov Chain Monte Carlo based approaches that account for the missing values, and the uncertainty they inherently introduce. Examples of these and other approaches to remedy the missing data problem can be found in many fields, including neuro-computing (Armitage and Lo, 1994; Meert, 1996), pattern recognition (Gabrys, 2002), and climatology (Jeffrey et al., 2001) to name a few.

Despite the clear theoretical shortcomings of simple imputation schemes (treated from different perspectives in Schafer, 1997; Armitage and Lo, 1994), the results in Chen et al. (2001) indicate that such simple imputation schemes combined with a neural network based traffic predictor, do produce accurate traffic predictions even when up to 30% of the input data to the model is missing. One (tentative) hypothesis may be that a (properly trained) neural network is robust to the "damage" caused by the imputation scheme applied. Slight changes in the statistical properties of the input data do not cause the neural network to produce inaccurate results. A possible explanation for this phenomenon is that the patterns formed by traffic measurements have very different statistical properties than the multivariate data sets used throughout (Schafer, 1997) which stem predominantly from medical statistics, and social sciences. Due to strong correlations through both time and space it makes sense that imputation with for example regression forecasts

[6] For example, closure of all lanes (zero flow + zero speed) or a complete empty road (no flow and no mean speed available).

yields sensible results for traffic measurements. Finally, for our particular application (en route travel time prediction) we seek *expected* (i.e. average) travel times, which may not be that sensible to a reduction in input variance due to simple imputation.

We therefore propose three strategies (S1–S3 in Table 1) to tackle input failure by means of three simple non-parameterized imputation schemes to be applied in the preprocessing layer. The first simple imputation method (strategy S1 in Table 1) considers spatial traffic patterns on the route of interest. If measurements on intermediate locations on the route are fraud we use spatial interpolation to fill in the gaps. The second simple imputation method (S2) is to consider time series of individual inputs only. Clearly, traffic measurements on fixed locations exhibit strong autocorrelation over time. Missing or corrupt measurements $U_d(t+1)$ from detector d at time instant $t+1$ can be replaced by a forecast $f_d(t+1)$ of a simple time series model, in our case by an exponentially moving average:

$$f_d(t+1) = f_d(t) + \alpha(U_d(t) - f_d(t)), \quad \alpha \in [0,1] \tag{7}$$

with α typically set to 0.3. The quantity $f_d(t)$ denotes the exponential forecast for input d at time instant t, while $U_d(t)$ denotes the dth input value at time instant t. Obviously, more complicated AR(I)MA models, multivariate regression models or non-linear forecasting techniques can be used, but these require offline design and calibration. Note that in case of structural detector failure, a time series forecast, such as (7), is not feasible, since it requires at least one valid measurement per input time series. Finally (strategy S3 in Table 1), we can combine both methods. In this case the imputed value is the minimum of the values obtained from the moving average (MA) and interpolation (Interp) method, imposing the maximum constraint (flow, speed) on the inputs.

4.3. Experimental setup

We have set up a network in FOSIM (Freeway Operations SImulation Model, see e.g. Vermijs and Schuurman, 1994) that resembles a 8.5 km stretch of the southbound carriage way of the A13 highway between two of the major Dutch cities in the western part of the Netherlands, The Hague and Rotterdam. This three-lane stretch contains four on- and five off ramps, and three weaving sections. The freeway stretch is divided into 12 sections, each enclosed between two consecutive detectors up- and downstream, measuring flows and harmonic time mean speeds[7] every 60 s. Fig. 6 gives a schematic drawing of the lay-out of the route.

The lay-out of the route provides sufficient information to set up a SSNN model for this particular freeway stretch. The hidden layer of the SSNN model contains 12 hidden neurons, each representing a section along the route, each receiving signals from those inputs associated with that section. The context layer also contains 12 units, while the output layer contains 1 neuron. The number of parameters (weights and biases) in the model then initially amounts to 228. Incidental (random) input failure is generated with a random generator J, producing numbers from a uniform distribution on [0, 1], such that each measurement from detector d at time t has an equal probability to be labeled corrupt. If the required level of corruption is set to 20% then a measurement is labeled corrupt if $J(d, t) \leqslant 0.2$. The maximum amount of incidental input failure consid-

[7] In real life, most inductive loop detectors record arithmetic time mean speeds. See Van Lint and Van der Zijpp (2003) for details.

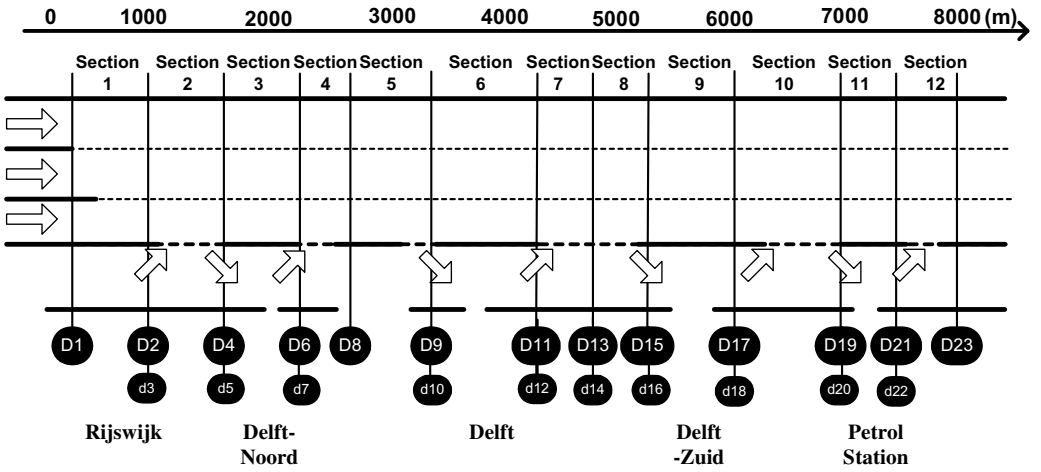

Fig. 6. Freeway stretch coded in microscopic traffic simulation model FOSIM. The stretch resembles a 8.5 km stretch of the the southbound A13 highway between The Hague and Rotterdam (The Netherlands).

ered is set at 40%. In case of structural detection failure ALL measurements $\{d, t\}$ from a specific detector d are labeled corrupt. Considering all possible combinations of failing detectors leads to a very large amount of test data, which is why we will only consider cases where 1, 3 or 5 detectors are structurally down on the relevant part of main carriage way, particularly the downstream detector on sections 4–9. This implies that the total amount of cases we consider add up to 16 test data sets.[8] We hypothesize that analyzing the effects of these failing detectors on the main carriage way gives enough insight into the effects and the possible solutions of structural input failure.

The following performance indicators are used in the ensuing:

MRE:	Mean relative error	$100\frac{1}{N}\sum\frac{e_n}{o_n}$
SRE:	St.dev. of relative error	$100\sqrt{\frac{1}{N-1}\sum\left(\frac{e_n}{o_n}-\frac{\mathrm{MRE}}{100}\right)^2}$
RMSEP:	Root mean squared error proportional	$\frac{100}{o}\sqrt{\frac{1}{N}\sum(e_n)^2}$

where N is the total number of observations, y_n the nth predicted value for the nth input, output pattern $\{u_n, o_n\}$, o the mean output value, and $e_n = y_n - o_n$ the prediction error for data pattern n.

[8] 1 detector down = 5 test data sets; 3 detectors down = $\frac{(5)!}{(5-3)!(3)!}$ = 10 test data sets; 5 detectors down = 1 test set; totalling in 16 test data sets.

4.4. Results

4.4.1. Incidental input failure

Table 2 presents the RMSEP performance (which is ratio between RMSE and mean travel time in percentages) for all proposed strategies on all available test data sets. The second row first shows that a SSNN model trained with 100% correct data is not able to handle missing data, which is to be expected. The *null* values replacing missing data are outside its input domain and its performance deteriorates steadily as the percentage of random failure in the test data sets increases. This strategy is provided as a baseline comparison for the other strategies.

The next four rows in Table 2 show that training the SSNN with increasing amounts of corrupted data does increase the robustness with respect to missing data but at a price in terms of performance. SSNN models trained with X percent of incidental input failure are capable of producing RMSEP values of 11–12% when tested against test data sets with similar incidental input failure levels (X percent). Obviously, to make the SSNN model robust, the amount of examples of input failure should be comparable to the amount expected in practice. The last three rows in Table 2 show the performance of an SSNN model trained with 100% correct data on increasingly corrupt test data sets in conjunction with three simple imputation strategies (Interpolation, Moving Average and a combination of these two respectively). Remarkably, all three outperform each of the S0 strategies, with the MA procedure (strategy S2) performing best. Even at 40% incidental input failure, the MA procedure reconstructs the data such that the SSNN model can still predict travel times as accurately as with 100% clean data (increase of RMSEP from 8% to 9% only). These results support the hypothesis that a (properly trained) neural network is robust to the "damage" caused by simple imputation schemes applied, that is, slight changes in the statistical properties of the input data do not cause the neural network to produce inaccurate output.

4.4.2. Structural input failure

Table 3 shows mean and standard deviation of the relative error and the RMSEP performance of the SSNN model against data sets in which respectively 1, 3 and 5 detectors on the congested part of main carriage way are structurally failing. Typically, null imputation (S0) strategies lead to underestimation, since the imputed signal (0.5) triggers the SSNN to a mean response. The simple imputation strategy (spatial interpolation) outperforms the (null imputation—S0) strategies in

Table 2
RMSEP (%) performance of all strategies applied to datasets with incidental input failure

Strategies	Test data sets				
	0%	10%	20%	30%	40%
S0 (0%)	8	19	33	52	77
S0 (10%)	11	10	14	18	26
S0 (20%)	13	12	11	12	15
S0 (30%)	18	15	13	12	12
S0 (40%)	23	20	17	14	12
S1 (Int)	8	8	9	10	12
S2 (MA)	8	8	8	8	9
S3 (MA/Int)	8	8	8	9	10

Table 3
Performance of SSNN travel time predictor on structural input failure of detectors on congested part

	Interp	0%	10%	20%	30%	40%
(*a*) *1 out of 5 down*						
MRE (%)	1	12	0	−4	−5	−7
SRE (%)	6	10	9	10	14	17
RMSEP(%)	9	17	11	13	17	21
(*b*) *3 out of 5 down*						
MRE (%)	1	39	20	7	1	0
SRE (%)	8	20	19	14	13	14
RMSEP(%)	10	53	20	15	15	17
(*c*) *5 out of 5 down*						
MRE (%)	1	93	62	31	15	13
SRE (%)	10	47	37	25	18	17
RMSEP(%)	14	72	50	30	22	20

which the SSNN is trained with (partial) structural detector failure. In the worst case scenario (five failing detectors) the spatial interpolation procedure allows the SSNN to still produce reasonable predictions with a relative error of 1% and a RMSEP of 14% (against 8% on clean data), as opposed to 13% and 20% respectively for the "best" S0 strategy (in which the SSNN is trained with 40% input failure).

4.5. Discussion of results

Due to the generality of the S0 strategies (SSNN models are trained with a mix of structural and incidental input failure), these more robust SSNN models offer a trade off between robustness and predictive accuracy. One could argue that it is possible to construct and train SSNN models for each particular structural detector failure problem and use these specifically trained SSNN models in each of those particular situations. This would, however, require a large number of SSNN models to be trained. In our case there are 8191 possible combinations[9] of structurally failing detectors on the main carriage way alone, yielding the same number of specialized SSNN models. On a route equipped with 25 detectors, this number increases to over three million combinations. Off course, this number could be significantly reduced by engineering judgement, for example by pre selection of likely combinations of failing detectors (based on history). This, however, does not guarantee a "waterproof" and hence robust framework.

Although it is clear there are many alternative imputation strategies that could be applied such as ARIMA models, neural networks, model based/Kalman filtering approaches and multiple

[9] There are 13 detectors on the main carriage way, yielding

$$\sum_{k=1}^{13} \frac{13!}{k!(13-k)!} = 8191$$

possible combinations of failing detectors.

imputation schemes, in both cases the principle of parsimony (which favors simple solutions as long as they perform well) leads us to conclude that simple non-parameterized imputation by means of spatial interpolation and a moving average offer a robust and easy to implement method for handling the missing data problem in the SSNN framework. These findings provide evidence that the SSNN model is robust to the 'statistical damage' done by the simple imputation procedures. There is enough redundancy in its parameter setting to allow partial distortion of its inputs without serious deterioration of its output. In this sense the proposed combination of SSNN and simple imputation provides for a framework that is robust with respect to both incidental and structural input failure, certainly at degrees of input failure that occur in real life.

5. Application of the travel time prediction framework in real-time

5.1. Real-time travel time prediction framework

In this final section we give preliminary results of the travel time prediction framework above applied in practice. The Regiolab Delft project (details can be found on the Regiolab-Delft Website, Van Zuylen and Muller, 2002) provides the ideal testbed for such an application. Within this project detailed traffic data is collected on a real-time basis from the road network covering the southwest of The Netherlands, including the A13 freeway corridor connecting The Hague and Rotterdam (see Fig. 7), which was used as a blueprint for the experiments based on synthetic data described earlier. The freeway data collection system MONICA provides 1 min aggregated traffic flows and 1 min averaged speeds from inductance loop detectors which are located about every 500 m on the A13 Highway.

The main drawback of applying a neural network based travel time prediction model in practice is the requirement of real (measured) travel times, since supervised neural networks such as the SSNN model learn by example, in this case data pairs $\{\mathbf{u}^{(t)}, o^{(t)}\}_{t=1}^{N}$ of observed speeds and flows and travel times. In the Regiolab area, no travel times are recorded on any of the freeway stretches yet, implying that we have to estimate these from the quantities we do measure: speeds. In Van Lint and Van der Zijpp (2003) a novel travel time estimation procedure, the Piece-wise Linear Speeds Based (PLSB) trajectory method is presented, which is a refinement of the Piece-wise Constant speeds Based (PCSB) trajectory method. On synthetic data (from the same source as used above) this procedure offers accurate travel time estimates with average estimation errors rarely larger than 1%. Although the PLSB method has not been validated against real data, extensive evaluation studies have been conducted on the PCSB method (e.g. Bovy and Thijs, 2000). Since the PLSB method is a refinement of the PCSB method, we are confident that these good results can be extrapolated to the PLSB method, since the results in Van Lint and Van der Zijpp (2003) indicate that the PLSB method yields further improvements in both bias and variance in comparison to the PCSB method.

5.2. Accounting for the effect of input failure on the offline travel time estimates

The effect of missing data on the performance and applicability of the SSNN framework in a real situation is (potentially) twofold. First, as with the experiments on synthetic data, it affects the

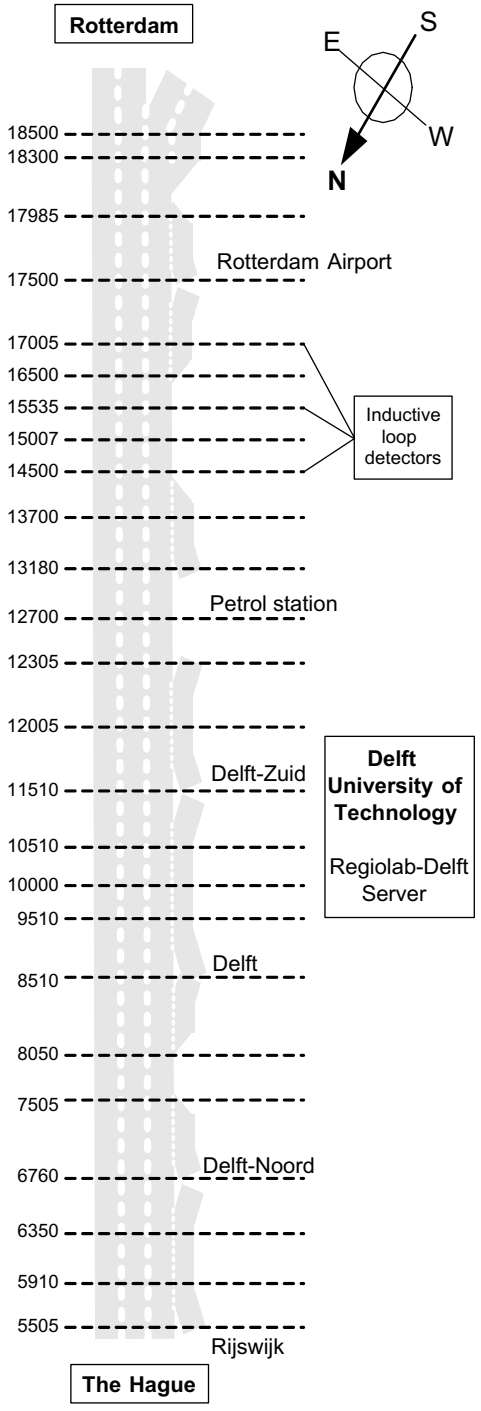

Fig. 7. A13 southbound freeway stretch between the Hague and Rotterdam. Note that for readability purposes this schematic view is not drawn to scale.

real-time operation of the SSNN, and second it also affects the training and testing procedure. In the latter case both input and target data (PLSB estimated travel times) are affected by missing data. In this case, applying repair algorithms may also bias the PLSB travel time estimates and result in learning the SSNN model "the wrong thing". In this section we briefly outline the effect of repairing input failure on the PLSB procedure.

In the piece-wise linear speed based (PLSB) trajectory algorithm (see Van Lint and Van der Zijpp, 2003) imaginary vehicles traverse through a grid in space and time. This grid is constituted of sections k enclosed by up- and downstream detectors and periods p, during which each of these detectors produces harmonic time averaged speeds of all passing vehicles. The PLSB algorithm then calculates the exit location and time of an (imaginary vehicle) entering a cell $\{k, p\}$ in space time as a convex combination of the harmonic time mean speeds measured at the up- and downstream end of section k during time period p. Given a fixed departure time t_0 at the most upstream detector, each vehicle trajectory is determined fully and provides an estimate for the mean travel time for vehicles departing at t_0.

In principle, neither sections nor periods need to be of a constant size (as long as for each region $\{k, p\}$ up- and downstream speeds are available), so that the easiest solution to dealing with missing data is by simply *omitting* these. The second solution we propose is to use the same imputation procedures as specified earlier, with the difference that now we can use interpolation in both the space and time direction, since the PLSB method is an offline method. Fig. 8 shows the RMSEP performance of the PLSB method on the same synthetic dataset as used above under both incidental (left graph) and structural detector failure (right graph). Surprisingly, in case of incidental failure, the interpolation routine repairs the underlying data such that up to 40% of missing data does not cause a serious increase in RMSEP. The other strategy (omitting missing

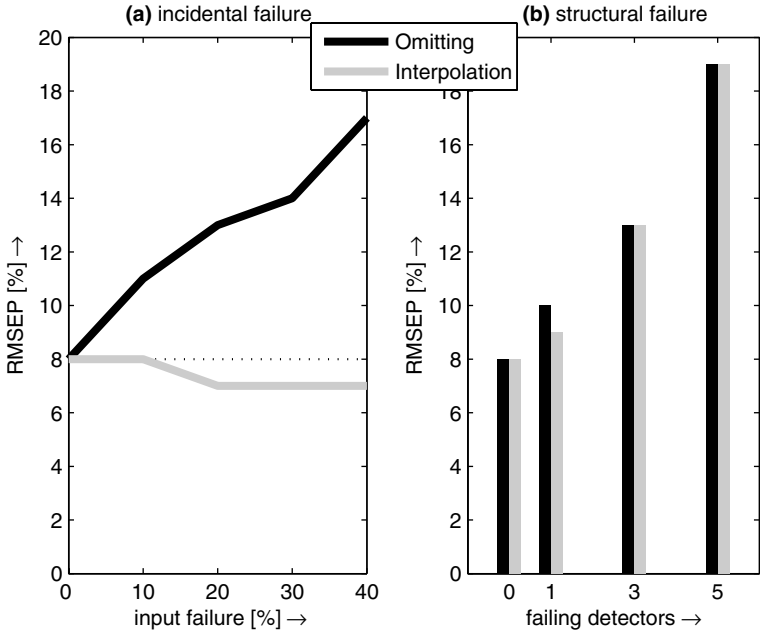

Fig. 8. Effect of dealing with incidental and structural input failure on performance of PLSB method.

data) does lead to deterioration in performance with increasing percentages of randomly missing data. In case of structural failure performance deteriorates for both repair strategies—Fig. 8(b) shows an increase from 8% to 18% RMSEP in case of five failing detectors. The (tentative) conclusion is that the interpolation routine works excellent for random failure and reasonable for structural failure. Detailed experimental research is still necessary to verify these results also on *real* data.

5.3. Data and input failure

In this articles three data sets are used, respectively data set A, B, and C. The first one is used for training the SSNN model and consists of 1 min flows and time averaged speeds from 19 inductive loop detectors on the southbound stretch of the A13 (Fig. 7), for the whole of 2000, yielding 525,600 records. All input data of data set A (speeds, flows) are cleaned and preprocessed by the offline preprocessing procedures (interpolation over space and time) described in the previous section. Also speeds were corrected for harmonic mean speed see Van Lint and Van der Zijpp (2003). Next, for each of the 525,600 departure time periods travel time estimates were generated using the PLSB method. Data set B is used for testing and contains data in the same format, but drawn from the first three months of 2001. The input data of this data set was preprocessed with the *online* method (i.e. a combination of an exponential filter and spatial interpolation) described earlier. Finally, data set C is a small data set in the same format as A and B, drawn from three days in June 2002 for which we collected actual travel times using camera's and license plate recognition software. On the average, in all three data sets the inductive loop data contain 12% input failure, which implies that nearly one out of eight measurements from the MONICA system are either missing or corrupt. On a minute-by-minute basis input failure is between 7% and 15% on most days, albeit their are some serious outliers of 20% and more. Unfortunately, on the three afternoon peaks during which actual travel times were recorded (data set C) serious input failure occurred of respectively 20%, 16% and 15%. There were numerous incidental input failures on a large number of detectors (cause unknown) while four main and three ramp detectors failed structurally on each of the three days. Particularly the failing detectors around 10 km and 11 km (Fig. 7 in the middle) may seriously affect the SSNN performance, since it is that area (near on and off-ramp Delft-Zuid) where the heaviest congestion occurs.

5.4. Results

In a real-time setting the application retrieves detector-data from the Regiolab webserver, preprocesses and cleans the data (online), and subsequently predicts travel times on a minute-by-minute-basis. On a Pentium IV (1.8 GHz/256 MB RAM) machine sequential prediction of 15 min, including preprocessing takes less than 0.5 s, making the model very suitable for deployment in real-time. Training the model on the same machine costs about 6.5 h, which is off course only required once (offline) before deployment. To compare the results of this model with the models currently operational on some of the travel time VMS panels on Dutch freeways, an online estimator was also fed with the same corrected data from the same time period.

Table 4 shows the results of a test run over data set B (compared to PLSB estimated travel times), and data set C (compared to real travel times). On data set B the SSNN produced a

Table 4
Performance of SSNN and instantaneous predictor on real data from inductive loops

	Dataset B (estimated travel times)		Dataset C (real travel times)	
	SSNN	Online estimator	SSNN	Online estimator
MRE (%)	1.6	8.9	4.8	17.7
SRE (%)	6.5	18.4	9.3	27.9
RMSE (s)	48	169	95.8	322.7

MRE of 1.5% and a standard deviation of the relative error of 6.5% outperforming the instantaneous predictor by far in terms of both bias and variance. On data set C, which is far smaller than data set B the results are worse, but still quite reasonable, certainly in the light of the effect of input failure on these three days discussed earlier. From these preliminary results, we argue the proposed combination of SSNN and simple imputation offers a robust and accurate travel time prediction framework, which still does a reasonable job even when up to one out of five measurements are not available.

6. Conclusions and further research topics

Accuracy and robustness with respect to missing or corrupt data are two key characteristics for any travel time prediction model that is to be applied in a real-time environment (e.g. for display on variable message signs (VMS's) on Freeways). We have presented a recurrent neural network topology, the so-called state-space neural network (SSNN), which yields accurate predictions and is insensitive to missing data, given these are imputed with simple algorithms such as spatial interpolation or exponential forecasting. We have tested two strands of approaches to deal with different kinds of input failure (incidental, structural) in conjunction with the SSNN model. The main findings are:

1. Although there are clear theoretical shortcomings to simple imputation schemes, their use may be justified in this particular application: the results indicate that the SSNN is robust to the "damage" done by naïve imputation schemes. This is true for both incidental (random) and structural input failure.
2. Inclusion of corrupted data in the training data sets of the SSNN model also improves robustness, but at the cost of predictive accuracy.
3. Results indicate that a combination of these imputation procedures and the SSNN model also provides satisfactory results in a real-time application, even when up to one out of five measurements are missing.

We conclude with to remarks. Firstly, in this article it is assumed that the underlying data collection system has a built-in facility for detecting missing or unreliable data, which in our test-site was in fact the case. The missing data problem in our case hence involves dealing with these gaps (through for example imputation) and accounting for their effect on the models outcome. In other data collection systems such a "fault-detection" mechanism may not be available, or insufficient

to effectively figure out which data are to be dubbed unreliable. We therefore strongly encourage further research into methodologies for data screening and checking. We finally remark that at the time of writing this manuscript two large scale evaluation studies have been taken up to further test the validity of the approach presented here on two test-sites in the Netherlands. The first, conducted on the same freeway stretch as reported in this article focusses on quantifying the reliability of the model's predictions by means of confidence intervals (Van Lint, 2004), while the second, which is due to finish mid 2005—concentrates on actual implementation of the framework on VMS panels on four freeway routes in the province of Utrecht, The Netherlands.

Acknowledgements

This research is part of the Regiolab-Delft Research Program (Van Zuylen and Muller, 2002) and is sponsored by the Dutch Ministry of Transport, Public Works and Water Management.

References

Abdel, A., Kitamura, R., Jovanis, P., 1997. Using stated preference data for studying the effect of advanced traffic information on drivers' route choice. Transportation Research Part C: Emerging Technologies 5C, 39–50.

Abdulhai, B., Porwal, H., Recker, W., 1999. Short term freeway traffic flow prediction using genetically-optimized time-delay-based neural networks. In: Proceedings of the 78th Annual Meeting of the Transportation Research Board. National Academies Press, Washington, DC, USA.

Armitage, W., Lo, J.-C., 1994. Enhancing the robustness of a feedforward neural network in the presence of missing data. In: Neural Networks, 1994. IEEE World Congress on Computational Intelligence, 1994 IEEE International Conference on, vol. 2, pp. 836–839.

Arnott, R., de Palma, A., Lindsey, R., 1991. Does providing information to drivers reduce traffic congestion? Transportation Research A 25 (5), 309–318.

Ben-Akiva, M., Bierlaire, M., Burton, D., Koutsopoulos, H.N., Mishalani, R., 2002. Network state estimation and prediction for real-time transportation management applications. In: Transportation Research Board Annual Meeting, CD-Rom. National Academies Press, Washington, DC, USA.

Bishop, C.M., 1995. Neural Networks for Pattern Recognition. Oxford University Press, United Kingdom.

Bovy, P.H.L., Thijs, R., 2000. Estimators of Travel Time for Road Networks, New Developments, Evaluation Results and Applications. Delft University Press, Delft, The Netherlands.

Chen, P.S.T., Srinivasan, K.K., Mahmassani, H.S., 1999. Effect of information quality on compliance behavior of commuters under real-time traffic information. In: Proceedings of the 77th Transportation Research Board Annual Meeting. National Academies Press, Washington, DC, USA.

Chen, H., Grant-Muller, S., Mussone, L., Montgomery, F., 2001. A study of hybrid neural network approaches and the effects of missing data on traffic forecasting. Neural Computing and Applications 10, 277–286.

Cheu, R.-L., 1998. Freeway traffic prediction using neural networks. In: Proceedings of the International Conference on Applications of Advanced Technologies in Transportation Engineering. ASCE, Reston, VA, USA, pp. 247–254.

Chun-Hsin, W., Chia-Chen, W., Da-Chun, S., Ming-Hua, C., Jan-Ming, H., 2003. Travel time prediction with support vector regression. In: Proceedings of the 2003 IEEE Conference on Intelligent Transportation Systems. IEEE, Shanghai, China.

D'Angelo, M.P., Al-Deek, H.M., Wang, M.C., 1999. Travel-time prediction for freeway corridors. Transportation Research Record 1676, 184–191.

Demuth, H., Beale, M., 1998. Neural Network Toolbox for Use with Matlab. The MathWorks Inc., USA.

Dia, H., 2001. An object-oriented neural network approach to short-term traffic forecasting. European Journal of Operational Research 131 (2), 253–261.

Dorffner, G., 1996. Neural networks for time series processing. Neural Network World 6, 447–468.
Dougherty, M., 1995. A review of neural networks applied to transport. Transportation Research C 3 (4), 247–260.
Elman, J., 1990. Finding structure in time. Cognitive Science 14, 179–211.
Foresee, F.D., Hagan, M.T., 1997. Gauss-newton approximation to Bayesian learning. In: International Conference on Neural Networks, vol. 3, pp. 1930–1935.
Gabrys, B., 2002. Neuro-fuzzy approach to processing inputs with missing values in pattern recognition problems. International Journal of Approximate Reasoning 30 (3), 149–179.
Hagan, M.T., Menhaj, M.B., 1994. Training feed-forward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks 5 (6), 989–993.
Hecht-Nielsen, R., 1990. Neurocomputing. Addison-Wesley Publishing Company, Readen, MA, USA.
Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Computation 9 (8), 1735–1780.
Hoogendoorn, S.P., Bovy, P.H.L., 2001. State-of-the-art of vehicular traffic flow modeling. Proceedings of the Institution of Mechanical Engineers 215 (1), 283–303.
Hu, T.Y., 2001. Evaluation framework for dynamic vehicle routing strategies under real-time information. Transportation Research Record 1774, 115–122.
Huisken, G., Van Berkum, E.C., 2003. A comparative analysis of short-range travel time prediction methods. In: TRB 2003 Annual Meeting CD-ROM. National Academies Press, Washington, DC, USA.
Innamaa, S., 2001. Short term prediction of highway travel time using mlp neural networks. In: Proceedings of the 8th World Congress on Intelligent Transport Systems, Sydney, Australia.
Ishak, S., Al-Deek, H., 2002. Performance evaluation of short-term time-series traffic prediction model. Journal of Transportation Engineering 128 (6), 490–498.
Ishak, S., Alecsandru, C., 2003. Optimizing traffic prediction performance of neural networks under various topological, input, and traffic condition settings. In: Transportation Research Board Annual Meeting CD-ROM. National Academies Press, Washington, DC, USA.
Ishak, S., Kotha, P., Alecsandru, C., 2003. Optimization of dynamic neural networks performance for short-term traffic prediction. In: Transportation Research Board Annual Meeting CD-ROM. National Academies Press, Washington, DC, USA.
Jeffrey, S.J., Carter, J.O., Moodie, K.B., Beswick, A.R., 2001. Using spatial interpolation to construct a comprehensive archive of Australian climate data. Environmental Modeling and Software 16 (4), 309–330.
Khattak, A.J., Schofer, J.L., Koppelman, F.S., 1995. Effect of traffic information on commuters propensity to change route and departure time. Journal of Advanced Transportation 29 (2), 193–212.
Kremer, S.C., 2001. Spatiotemporal connectionist networks: a taxonomy and review. Neural Computation 13, 249–306.
MacKay, D.J.C., 1995. Probable networks and plausible predictions: a review of practical Bayesian methods for supervised neural networks. Network: Computation in Neural Systems 6 (3), 469–505.
Mahmassani, H.S., Liu, Y.-H., 1999. Dynamics of commuting decision behavior under advanced traveller information systems. Transportation Research C 7, 91–107.
Meert, K., 1996. A real-time recurrent learning network structure for dealing with missing sensor data. In: Proceedings of the 1996 IEEE Conference on Neural Networks, Washington, DC, USA, vol. 3, pp. 1600–1605.
Nguyen, D., Widrow, B., 1990. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In: Proceedings of the International Joint Conference on Neural Networks, vol. 3, pp. 21–26.
Park, D.J., Rilett, L.R., 1998. Forecasting multiple-period freeway link travel times using modular neural networks. Transportation Research Record 1617, 163–170.
Park, D., Rilett, L., 1999. Forecasting freeway link travel times with a multilayer feed-forward neural network. Computer Aided Civil and Infrastructure Engineering 14 (5), 357–367.
Rice, J., Van Zwet, E., 2001. A simple and effective method for predicting travel times on freeways. In: Proceedings of the IEEE Conference on Intelligent Transportation Systems, Oakland, CA, United States, pp. 227–232.
Rilett, L.R., Park, D., 2001. Direct forecasting of freeway corridor travel times using spectral basis neural networks. Transportation Research Record 1752, 140–147.
Sarle, W.S., 2004. Neural network faq, ftp://ftp.sas.com/pub/neural/faq.html, monthly posting to the Usenet newsgroup comp.ai.neural-nets.
Schafer, J.L., 1997. Analysis of Incomplete Multivariate Data. Chapman and Hall, London, UK.

Smulders, S.A., Messmer, A., Knibbe, W.J.J., 1999. Real-time application of metanet in traffic management centres. In: Proceedings of the 6th World Congress on Intelligent Transport Systems (ITS), Toronto, Canada.

Sun, H., Liu, H.X., Xiao, H., He, R.R., Ran, B., 2003. Short-term traffic forecasting using the local linear regression model. In: Transportation Research Board Annual Meeting CD-ROM. National Academies Press, Washington, DC, USA.

Van Berkum, P., Van der Mede, P., 1993. The Impact of Traffic Information: Dynamics in Route and Departure Time Choice. Delft University Press, Delft, The Netherlands. PhD thesis of the Transport and Planning Department, Faculty of Civil Engineering and Geosciences, Delft University of Technology.

Van Grol, R., Middelham, F., van Ruremonde, A., 1997. Daccord/boss: its developments in the Amsterdam region. In: Mobility for Everyone, Proceedings of the 4th World Congress on Intelligent Transport Systems. ITS Congress Association, Berlin, paper no. 2164.

Van Lint, J.W.C., 2004. Reliable Travel Time Prediction for Freeways. TRAIL thesis series. TRAIL Research School, Delft.

Van Lint, J.W.C., Hoogendoorn, S.P., Van Zuylen, H.J., 2002. Freeway travel time prediction with state-space neural networks—modeling state-space dynamics with recurrent neural networks. Transportation Research Record 1811, 30–39.

Van Lint, J.W.C., Van der Zijpp, N.J., 2003. Improving a travel time estimation algorithm by using dual loop detectors. Transportation Research Record 1855, 41–48.

Van Toorenburg, J.A.C., 1998. Astrival functionele specificatie algoritme, rijtijd en filelengteschatter voor meetvak (in dutch), Technical report, AVV Transport Research Centre, Ministry of Transport, Public Works and Water Management.

Van Zuylen, H.J., Muller, T.H.J., 2002. Regiolab Delft. In: Proceedings of the 9th World Congress on Intelligent Transport Systems, CD-Rom, Chicago, IL, USA. Available from: <http://www.regiolab-delft.nl>.

Vermijs, R.G.M.M., Schuurman, H., 1994. Evaluating capacity of freeway weaving sections and on-ramps using the microscopic simulation model fosim. In: Proceedings of the second international symposium on highway capacity, Sydney, Australia, vol. 2, pp. 651–670.

Williams, B.M., 2001. Multivariate traffic flow prediction: an evaluation of arimax modelling. In: Transportation Research Board 80th Annual Meeting, CD-Rom. National Academies Press, Washington, DC, USA.

Zhang, X., Rice, J.A., 2003. Short-term travel time prediction. Transportation Research Part C: Emerging Technologies 11 (3–4), 187–210.